

This Page Is Inserted by IFW Operations
and is not a part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

**As rescanning documents *will not* correct images,
please do not report the images to the
Image Problem Mailbox.**

SEEDEX: A MAC protocol for ad hoc networks*[†]

R. Rozovsky and P. R. Kumar

Dept. of Electrical and Computer Engineering, and
Coordinated Science Laboratory
University of Illinois
1308 West Main Street
Urbana, IL 61801
USA

ABSTRACT

Motivated by the poor experimental scaling reported in a study of the performance of ad hoc networks in [15], we propose a new protocol for media access control in ad hoc networks. Our protocol seeks to avoid collisions without making explicit reservations for each and every packet. The key idea is to employ a random schedule which is driven by a pseudo-random number generator. By exchanging the seeds of their pseudo-random number generators within a two-hop neighborhood, the nodes effectively publish their schedules to all hidden as well as exposed nodes. This allows each node to opportunistically choose transmission slots. This scheme can also be employed during the reservation phase of a protocol such as IEEE 802.11. Throughput calculations and simulation results are presented.

1. INTRODUCTION

A key property that distinguishes the wireless radio medium from wireline is that it is a shared medium. Thus, assuming that neighboring nodes are within range of each other, in Figure 1 we see that only certain sets of simultaneous successful transmissions are feasible. When node *C* transmits to node *D*, node *A* cannot successfully transmit a packet at the same time to node *B* since *C*'s transmission causes a collision at *B*. Thus, nodes need to coordinate their transmissions in order to communicate. However, such coordi-

*This material is based upon work partially supported by the U.S. Army Research Office under Contracts DAAD19-00-1-0466 and DAAD19-01010-465, the Office of Naval Research under Contract N00014-99-1-0696, and DARPA under Contract No. N00014-01-1-0576. Any opinions, findings, and conclusions are those of the authors and do not necessarily reflect the views of the above agencies.

[†]Please address all correspondence to the last author at University of Illinois, Coordinated Science Laboratory, 1308 West Main Street, Urbana, Illinois 61801, USA. Email: prkumar@uiuc.edu. Web: black.csl.uiuc.edu/~prkumar.

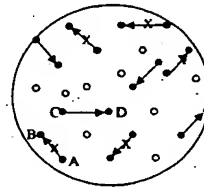


Figure 1: Only certain sets of transmissions can be simultaneously successful.

nation can only be achieved through communication. Thus communication needs coordination which in turn needs communication. Note also that nodes may not know when other nodes have packets to transmit. This gives rise to the fundamental Media Access Control problem for ad hoc networks: How should nodes make decisions in real time on when to transmit packets?

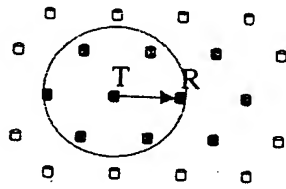
2. THE IEEE 802.11 PROTOCOL

One solution, which is available in many products such as Lucent's WaveLan Cards, CISCO's Aironet Cards, etc., is the IEEE 802.11 Protocol (see [1] and [3]). This employs a four-way handshake for each DATA packet. Consider the situation shown in Figure 2.

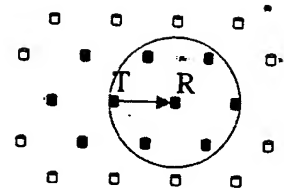
Suppose node *T* has a packet to send to a neighboring node *R*. Then it first sends a RTS (request-to-send) packet. This is heard by all packets in the neighborhood of *T*, including *R* (assuming they experience no conflict). The neighbors of node *T* which hear this RTS are then supposed to stay silent for a while. Upon hearing the RTS, node *R* sends back a CTS (clear-to-send) packet. This is heard by node *R*'s neighbors (again assuming they experience no conflict), and they too have to then stay silent for a while. Since node *T*'s neighbors have been silenced, node *T* experiences no conflict, and can hear node *R*'s CTS. Upon hearing the CTS, node *T* sends its DATA packet. This is successfully received by node *R* since node *R*'s neighbors were earlier silenced by its CTS packet. After receiving the DATA packet, node *R* sends back an ACK, which is again received successfully by *T*. After this, the neighborhoods of *R* and *T* are released from their silence.

One feature to note is that two neighborhoods (of *T* and of

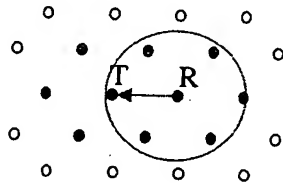
RTS - Neighbors of Transmitter are silenced



Data is sent



CTS - Neighbors of Receiver are silenced



ACK is returned

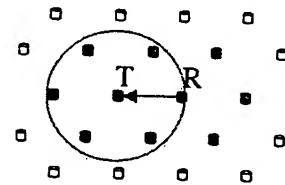


Figure 2: The RTS-CTS-DATA-ACK handshake of IEEE 802.11.

R) are silenced. This is wasteful since only the receiver's neighborhood has to stay silent in order for R to successfully receive the DATA packet from T (the so called "hidden terminal" problem). Moreover, the elaborate four way RTS-CTS-DATA-ACK handshake is necessary for each and every packet, which again can be wasteful. Finally, whenever a collision occurs, nodes employ a "backoff" mechanism as in ALOHA (see [3]). This again can be wasteful.

Indeed, a scaling experiment conducted on a network ranging from 2 to 12 nodes, reported in [15] showed that the per node throughput declined as $O(\frac{1}{n^{1.88}})$ bits/sec. This scaling law is considerably worse than the optimal scaling law $O(\frac{1}{\sqrt{n \log n}})$ bits/sec shown to be feasible in [14]. Indeed it is worse than even the throughput of $O(\frac{1}{n})$ bits/sec that is feasible by even when the nodes are colocated.

This has motivated us to develop a new protocol for the MAC layer. This protocol, which we call SEEDEX, attempts to make reservations without explicitly making them, as we describe in the sequel.

Now we present a brief review of the literature. To address the issue of efficient and fair allocation of the bandwidth among stations in the presence of the hidden terminal problem, the MACAW protocol [16] introduces a more complex RTS-CTS-DS-DATA-ACK message exchange and a gentler adjustment backoff mechanism. FAMA [12] guarantees collision-free transmission of one or more data packets, using carrier sensing and collision avoidance to assign a station control of the channel. The RTS part of the handshake is removed in MACA-BI protocol, which is shown in [7] to

be more robust to control packet collisions and a finite turnaround time problems. Efficiency of the contention access at low loads and stability of the allocation-based access are exploited in the protocols combining the two methods of access. HRMA [18], CHMA [2], and MACA-CT [13] use reservation mechanisms with frequency-hopping spread-spectrum, while ADAPT [5], ABROAD [6], CATA [19], FPRP [4] are based on contention for or within TDMA slots. A control channel with transmit and receive busy tones is used in DBTMA scheme [17] for RTS/CTS dialogue to improve the data channel utilization.

Closest to our approach are [10] and the sequence of [8], [9]. [10] presents a link layer protocol, called Adaptive Receive Node Scheduling (ARNS), for a multiple satellite network. ARNS employs a pseudo-random time line to compute receiver schedules, and provides each satellite with a schedule for its neighboring satellites so that the intended receiver's antenna is pointed to the transmitter and it is listening for a transmission, thus avoiding contention. Another set of works close to our approach is [8], [9], where pseudo-random scheduling is proposed for fair, low-delay energy-conserving multiple access in one-cell identification networks environment.

3. IF WE ONLY KNEW THE SCHEDULES OF ALL NODES IN A TWO HOP NEIGHBORHOOD

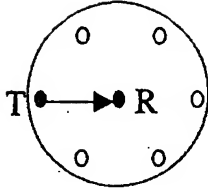


Figure 3: Node T can send a packet to node R without a collision since node R as well as all its neighbors are guaranteed to stay silent.

Suppose that all nodes could publish their schedules. By this, we mean a statement of the following form:

- 0 ms - 1 ms: Silent, listening for packets, called state " L "
- 1 ms - 2 ms: Possibly send a packet, called state " PT "
- 2 ms - 3 ms: Possibly send a packet (PT)
- 3 ms - 4 ms: Silent and listening for packets (L)

Suppose that a node T knows the schedules of all the nodes in a two hop neighborhood of itself. Then, if node T wishes to send a packet to its neighbor R , it could choose a slot when

1. Node T is in state PT , i.e., it has announced it may possibly send a packet.
2. Node R is in state L , i.e., it has announced it will stay silent.
3. All of node R 's neighbors are in state L , i.e., they have announced that they will stay silent.

Then, as shown in Figure 3, node T can successfully send a packet to node R without fear of a collision at R .

4. CHOOSING A RANDOM SCHEDULE

The first question that arises is: How do we choose a schedule? We will choose a *random schedule*. Each node chooses a probability parameter $0 < p < 1$. With probability p it will mark a slot as being one where it may "possibly transmit" a packet (state PT), and otherwise (with probability $(1 - p)$) it will stay "silent" (state L). This is done independently from slot to slot, as shown in Figure 4. Thus a schedule could simply be an i.i.d. Bernoulli sequence.

A more complicated schedule can be generated by driving a Finite State Machine (FSM) with a pseudo-random number generator. One can simply label the states of the FSM with either S (for Silent) or PT (for possible transmit), as shown in Figure 5. This is analogous to a Markov chain, and allows

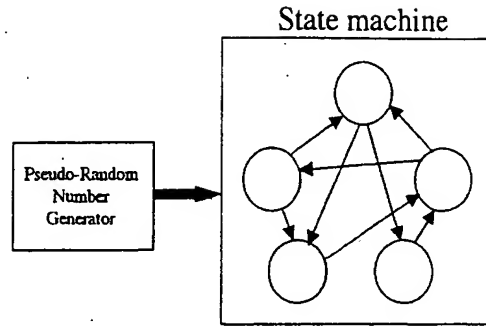


Figure 5: Driving a Finite State Machine with a pseudo-random number generator, to create a random schedule.

for some temporal correlations between neighboring slots, which may be advantageous in reducing delays.

We will fix our attention in this paper though to the simpler case of an i.i.d. Bernoulli sequence.

5. THE CENTRAL IDEA OF SEEDEX: PUBLISHING RANDOM SCHEDULES BY EXCHANGING SEEDS

Consider the i.i.d. Bernoulli schedule, as in Figure 4. It is generated through the use of a pseudo-random number generator. Such pseudo-random number generators have an internal state, whose initial value is called the "seed." A sequence of random looking numbers which define the schedule is then generated through a recurrence equation. Thus, if node A knows the initial seed of the pseudo-random number generator used by node B , then node A can determine node B 's schedule.

This leads to a key idea: Nodes simply have to publish their seeds, and not their entire schedules.

Note that a node needs to let all other nodes in a two-hop neighborhood of itself know what its seed is. This can be done through a fan-in and fan-out procedure, as shown in Figure 6.

Every node broadcasts the seeds of all its neighbors that it knows about, including itself, to all its neighbors (fan-out). After hearing a similar broadcast from each of its neighbors (fan-in), it then again broadcasts the seeds of all its neighbors to all its neighbors. Seeds are thus exchanged with all nodes in a two-hop neighborhood.

To cope with mobility and nodes entering or leaving a neighborhood, this procedure of broadcasting all the seeds of its neighbors could be repeated periodically. Second, a node should broadcast not the initial condition of the random number generator of a neighbor, which may have occurred at some indeterministic time in the past, but the current state of the pseudo-random number generator. Note that every node keeps track of the current state of its neighbors by simply propagating the recurrence equation. This noti-

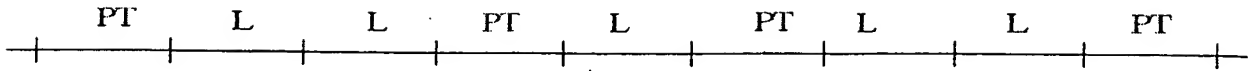


Figure 4: A random schedule given by a Bernoulli sequence.

Send SEED to your neighbor

Neighbor sends all SEEDs to you

Now you know SEEDs of all your 2-hop neighbors

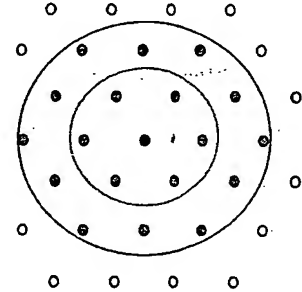
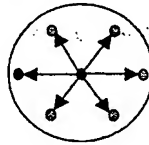
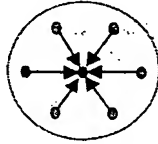


Figure 6: The fan-out and fan-in procedure for exchange seeds in a two-hop neighborhood.

fication of the current state obviates the need to tell other nodes what the initial times were. The periodic repetition of information is also healthy since nodes can correct their perceptions of the states of the pseudo-random generators if errors have crept in for some reason in since the last update. Last, if nodes enter or leave the neighborhood, then this repetition updates all other nodes within a two-hop neighborhood of the occurrence.

We should note that if the scheme involving Finite State Machines (rather than simple Bernoulli random variables) is used, then a node will also have to transmit the state of the Finite State Machine in addition to the state of the pseudo-random number generator.

6. WHEN DOES A NODE TRANSMIT A PACKET?

Suppose a node T has a packet to transmit to a neighboring node R . When should it transmit it?

First, the node T should wait for a slot at which simultaneously node T is in a "Possibly Transmit" state and node R is in a "Listen" state. At such a slot, node T may discover that there are n other nodes of R which are also in a "Possibly Transmit" state. Suppose, as in Figure 7, that there are $n = 2$ other neighbors of node R which are also in the "Possibly Transmit" state. Then node T should transmit with the probability $\text{Min} \left\{ \frac{\alpha}{n+1}, 1 \right\}$, and refrain from transmitting its packet in that slot with the complementary probability $1 - \text{Min} \left\{ \frac{\alpha}{n+1}, 1 \right\}$.

This rule is arrived at through the following reasoning. Suppose all the other $n = 2$ nodes have a packet to send to R (which, as we will discuss in the next paragraph, need not be the case). Then if each of the $(n + 1)$ nodes transmits with probability π , the probability that there will be exactly one

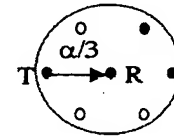


Figure 7: Node transmits with probability $\frac{\alpha}{3}$.

successful reception is $(n + 1)\pi(1 - \pi)^n$, which is maximized when $\pi = \frac{\alpha}{n+1}$ with $\alpha = 1$.

Note however that all the other n neighbors of R which are in a "Possibly Transmit" state may not actually have a packet to transmit. Thus, node T can afford to be more aggressive, and transmit with probability $\frac{\alpha}{n+1}$ where $\alpha \geq 1$. This motivates the use of the parameter α . In light traffic α should be large, while in heavy traffic α should be low. In our experiment described in Section 9, we found the choice $\alpha \approx 2.5$ optimal in light traffic, and $\alpha \approx 1.5$ optimal in heavy traffic. Note also that to avoid probabilities larger than one, the "Min" operation is introduced to give the expression $\text{Min} \left\{ \frac{\alpha}{n+1}, 1 \right\}$.

One more point to note is that the other neighbors of R which may be in a "Possibly Transmit" state may have a packet to send to another neighbor different from R , as shown in Figure 8.

Then, while node T notes that there are two other neighbors of R in a possibly transmit state, and so sends its packet with probability $\frac{\alpha}{3}$ (assuming $\alpha < 3$), node T' looks at the neighborhood of its intended recipient R' , and since that contains three other nodes in a possible transmit state, it transmits with probability $\frac{\alpha}{4}$.

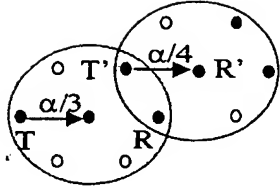


Figure 8: Node T wants to send a packet to R , and Node T' wants to send a packet to R' .

Thus, not all neighbors of node R in a "Possibly Transmit" state need transmit with the same probability. Nevertheless, due to the absence of information on when a node has a packet to transmit, and to whom, the guideline of transmitting with probability $\text{Min} \left\{ \frac{\alpha}{n+1}, 1 \right\}$ will be employed.

7. WHAT IS A GOOD CHOICE OF p ?

Note that each node stays "Silent" on a slot with probability $(1-p)$, and is in a "Possibly Transmit" state with probability p . What is a good choice of p ?

This can be analyzed using the approximation that all neighbors of R also have packets to send to R whenever they are in a "Possibly Transmit" state.

Suppose node R has N neighbors. Then node T successfully transmits a packet to node R on a slot when (i) node T is in the "Possibly Transmit" state, which occurs with probability p , (ii) node R is in the "Listen" state, which occurs with probability $(1-p)$, (iii) j other neighbors ($0 \leq j \leq N-1$) are also in a "Possibly Transmit" state, and the remaining $(N-1-j)$ neighbors of R are in a "Silent" state, which happens with probability $\binom{N-1}{j} p^j (1-p)^{N-1-j}$, (iv) for each such value of $j = 0, \dots, N-1$, only node T decides to send a packet, which happens with probability $\frac{1}{j+1}$, while the other j nodes all decide not to send a packet to R , which occurs with probability $\left(1 - \frac{1}{j+1}\right)^j$. Thus the probability of successful transmission of a packet from T to R on a slot, denoted λ_{TR} , is

$$\lambda_{TR} = p(1-p) \sum_{j=0}^{N-1} \binom{N-1}{j} p^j (1-p)^{N-1-j} \frac{1}{j+1} \times \left(1 - \frac{1}{j+1}\right)^j.$$

Noting that there are a total of $N+1$ nodes in the wireless footprint, i.e., within range, let us define the "throughput" (or channel utilization) of the scheme as $\lambda := (N+1)\lambda_{TR}$.

For $N = 6$, this expression λ is maximized (see Figure 9) when $p = 0.246$. Simulation results show that the maximizing value is $p \approx 0.21$, and that it is quite insensitive to the traffic load, see Figure 10. Our simulation experience shows that it appears to be relatively insensitive even to the topology.

8. ACKS

When a node T transmits a packet intended for R , it has no guarantee that R indeed receives the packet successfully.

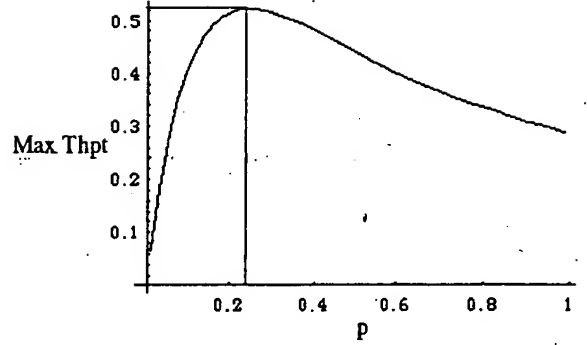


Figure 9: A plot of λ versus p .

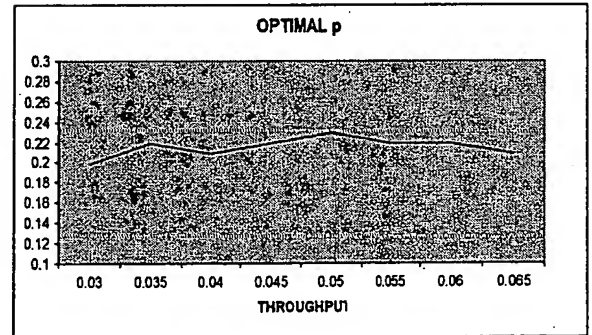


Figure 10: A plot of the maximizing p for various throughputs, obtained from simulation.

This is due to several reasons. First, the wireless medium is itself unreliable due to the presence of obstacles, shadowing, multipath effects, fading, etc. Second, the packet may collide at R with another packet being transmitted by a neighbor of R which is "hidden" from T . Thus, for services needing reliable transport, we believe that link level acknowledgments are a must in ad hoc networks.

When should R send an ACK, and what particular packet of T (a la TCP) should it ACK? First, since our scheme is using synchronized slots, we can simply set aside a small time at the end of the slot carrying the DATA packet from T to R to send an ACK back from R to T . Then R can either ACK the particular packet received, or NACK the "next awaited packet."

9. SOME PERFORMANCE NUMBERS

Our first simulation experiment, conducted on NS, consists of 100 nodes located at the vertices of a regular hexagonal tessellation. Each node chooses a random neighboring recipient for each packet.

We also wish to study the effect of channel errors on the performance of our scheme. (Note that channel errors can have adverse impact on a scheme making "reservation," since they can disrupt such reservations). To study the effect of channel errors, we simply choose a probability of error for each packet, which is then applied independently for each packet.

We plot below the throughput versus delay characteristic in Figure 11. We exhibit the throughput at which packets move from a node T to a neighboring node R . The values are averaged over the 55 nodes in the center of the network. We show the performance for six different levels of per packet channel error errors, 0%, 1%, 2%, 3%, 4%, and 5%. The larger delays in the figure are for higher channel error probabilities. The throughput ν is calculated as $3 \times (\text{Packets per second per flow})$, and the delay D is measured in slots. (See [11] for an explanation of the normalization used).

One may note that the performance of the scheme only degrades softly in the presence of channel error.

10. USING SEEDER FOR RTS RESERVATIONS

We can further enhance the SEEDER protocol as follows.

The idea is to employ a hybrid, using SEEDER only on the RTS packets which are used to make reservations. The CTS then follows, followed in turn by a DATA and an ACK, just as in IEEE 802.11. This has several advantages. First, collisions are avoided for the long DATA packets since their slots are "reserved." The only contention for slots is by the RTS packets which are short. This contention is resolved through SEEDER. This allows for a more efficient utilization of the channel since it tries to avoid collisions of the larger DATA packets. There is another advantage in using SEEDER for RTS packets, as opposed to "ALOHA" type schemes or carrier sensing schemes, such as used in IEEE 802.11. The backoff counters do not migrate to large values, as in IEEE 802.11, which we suspect could be one cause for the very poor throughput measured in experimental scaling

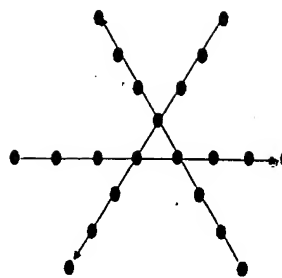


Figure 13: Three intersecting flows.

in [15].

We call this scheme SEEDER-R, for SEEDER with Reservations.

11. SEEDER-R: SEEDER WITH RESERVATIONS

The full SEEDER-R scheme which employs RTS-CTS-DATA-ACK, with RTS contending via SEEDER, is as follows. The RTS, CTS, and ACK packets are each 25 bytes long, while DATA packets are 1000 bytes long.

A node T contends for an RTS slot via SEEDER. This is successfully received by R . R sends a CTS to T on the next slot. Then T sends a DATA packet. This is followed by an ACK packet from R . After this, another contention period for RTS follows. Figure 12 illustrates the operation.

12. PERFORMANCE COMPARISON OF SEEDER-R WITH IEEE 802.11

We have compared the performance of SEEDER-R with IEEE 802.11 on a network with three intersecting flows, as shown in Figure 13, in order to illustrate its performance in an environment with contention.

The throughput versus mean delay, and throughput versus standard deviation of delay, are shown in Figures 14 and 15. As earlier, for the throughput, we display $N + 1 = 3$ times the total of the throughput rates of the three flows, which is an indicator of channel utilization in the congested neighborhood.

We note that the capacity, i.e., the maximum throughput that can be provided, is about 10% greater than that obtainable from IEEE 802.11.

The mean delay is relatively constant and lower than that of IEEE 802.11 by 40%, while the standard deviation of delay (delay jitter) is substantially reduced by a factor of about five.

13. HOW CAN ONE PROVIDE QOS?

Can we provide different levels of throughput for different flows? We show in this section how this may be done.

The key idea is to adjust the value of p that a node chooses. Let us denote by p_i , the value of Prob (Possibly Transmit) that node i uses.

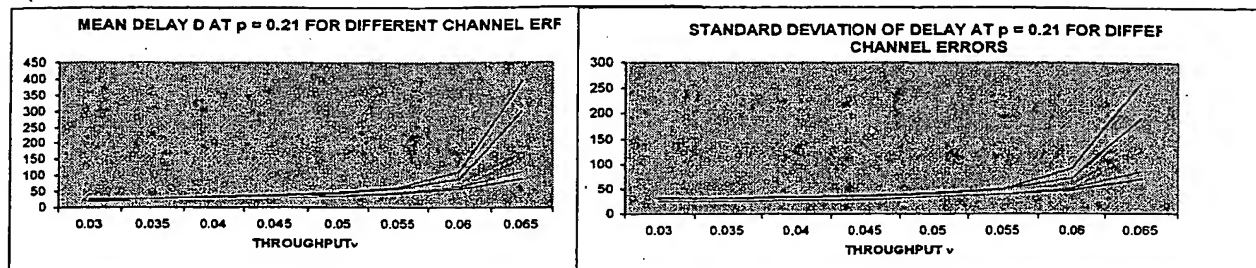


Figure 11: Throughput versus Mean and Standard Deviation of Delay for SEEDEx. Shown is the performance for six different levels of per packet channel errors: 0%, 1%, 2%, 3%, 4%, and 5%. The larger values of delays in the figure are for higher channel error probabilities.

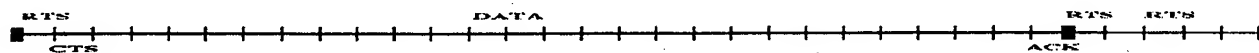


Figure 12: The operation of SEEDEx-R.

Throughput	SEEDEx	IEEE 802.11
0.2	15.52	24.34
0.3	15.74	21.56
0.4	15.50	20.34
0.5	15.54	24.04
0.55	15.64	30.13
0.6	33.63	809.09

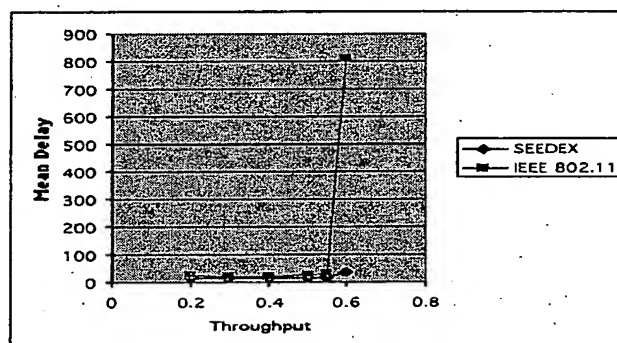


Figure 14: Throughput versus Mean Delay for SEEDEx-R and IEEE 802.11.

Throughput	SEEDEx	IEEE 802.11
0.2	2.85	18.68
0.3	3.08	13.61
0.4	2.90	11.59
0.5	2.97	15.54
0.55	3.29	21.01
0.6	18.93	748.77

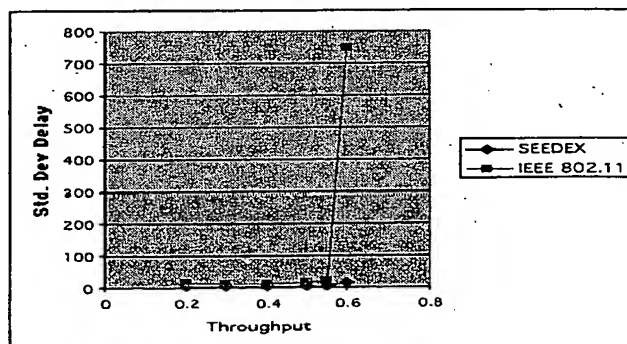


Figure 15: Throughput versus Standard Deviation of Delay for SEEDEx-R and IEEE 802.11.

We now show that the p_i 's can be adjusted to vary the throughput obtained. Consider a scenario with Node 0 surrounded by nodes, $1, 2, \dots, N$ in its one-hop neighborhood: Then, by a straightforward calculation, the service rate μ_1 that node 1 obtains for its packets to node 0, is

$$\mu_1 = p_1(1-p_0) \left[\sum_{0 \leq k_2 \leq 1} \prod_{i=2}^N p_i^{k_i} (1-p_i)^{1-k_i} \right] \times \frac{1}{1 + \sum_{i=2}^N k_i} \left(1 - \frac{1}{1 + \sum_{i=2}^N k_i} \right)^{\sum_{i=2}^N k_i}$$

By using Jensen's inequality, this is lower bounded as follows:

$$\mu_1 \geq p_1(1-p_0) \frac{1}{1 + \sum_{i=2}^N p_i} \left(\frac{\sum_{i=2}^N p_i}{1 + \sum_{i=2}^N p_i} \right)^{\sum_{i=2}^N p_i} \geq \frac{p_1(1-p_0)}{1.4 + e \sum_{i=2}^N p_i}$$

The last inequality follows from $\left(\frac{1}{1+y}\right) \left(\frac{y}{1+y}\right)^y \geq \frac{1}{1.4+ey}$.

One can repeat this argument for the other nodes, and deduce that

$$\sum_{i=1}^N \mu_i \geq \frac{(1-p_0) \sum_{i=1}^N p_i}{1.4 + e \sum_{i=2}^N p_i}$$

Now we show how to allocate the p_i 's to provide differential QoS. Suppose that two guidelines are followed:

- (i) $0 < p_i \leq \bar{p} < 1$ for all $i = 0, 1, \dots, N$.
- (ii) $\sum_{i=1}^N p_i \geq P$.

Then it is easy to see that

$$\frac{\mu_i}{p_i} \geq \frac{1-P}{1.4 + e(N-1)\bar{p}}$$

Thus, increasing p_i increases μ_i (up to a limit), and provides a guideline for providing different throughputs for different flows and can therefore be used to control QoS. We refer the reader to [11] for more details.

Finally, we note that SEEDEX can also be used in a multi-cast environment since a transmitter knows the states of all its two-hop neighbors.

14. CONCLUDING REMARKS

The SEEDEX Protocol is motivated by the goal of improving the scaling performance of ad hoc networks. It seeks to avoid making reservations for each and every packet, and also does not require silencing the neighborhoods of both

the receiver as well as transmitter. It also does not employ backoff counters in the case of collisions.

Several issues such as overhead, the fan-in procedure, correlations between slots, adaptation of α , impact of topology, etc., are worthy of further investigation.

As an initial foray, and as a proof of concept, we have currently implemented the scheme using some off the shelf hardware: Cisco Aironet cards on laptops running Linux. Significant challenges included working around the carrier sensing mechanism, and the synchronization of slots of the laptops. To achieve these goals, capacity is intentionally sacrificed. The next phase is to conduct some larger scale testing. The availability of synchronized slots, as in Bluetooth hardware, would be a big advantage.

15. REFERENCES

- [1] IEEE Protocol 802.11. *Draft standard for wireless LAN: Medium access control (MAC) and physical layer (PHY) specifications*. IEEE, July 1996.
- [2] A.Tzamaloukas and J.J.Garcia-Luna-Aceves. Channel-hopping multiple access. In *Proceedings of the IEEE International Conference on Computer Communication and Network (IC3N '00)*, Las Vegas, Nevada, October 2000.
- [3] IEEE Computer Society LAN MAN Standards Committee. *Wireless LAN medium access control (MAC) and physical layer (PHY) specifications*. IEEE Standard 802.11-1997, The Institute of Electrical and Electronics Engineers, New York, NY, 1997.
- [4] C.Zhu and S.Corso. A five-phase reservation protocol (FPRP) for mobile ad hoc networks. In *Proc. of IEEE INFOCOM*, New York, NY, 1998.
- [5] I.Chlamtac et al. ADAPT: A dynamically self-adjusting media access control protocol for ad hoc networks. In *Proc. of IEEE GLOBECOM*, pages 11-15, December 1999.
- [6] I.Chlamtac et al. An adaptive medium access control (MAC) protocol for reliable broadcast in wireless networks. In *IEEE International Conference on Communications*, New Orleans, June 2000. ICC.
- [7] F.Talucci, M.Gerla, and L.Fratta. MACABI (MACA by invitation): A receiver oriented access protocol for wireless multiple networks. In *PIMRC '97*, pages 1-4, Helsinki, Finland, September 1994.
- [8] I.Chlamtac, C.Petrioli, and J. Redi. An energy conserving access protocol for wireless communication. In *IEEE-ICG97*, pages 1059-62, Montreal, 1997.
- [9] I.Chlamtac, C.Petrioli, and J. Redi. Extensions to the pseudo-random class of energy-conserving access protocols. In *IEEE Int. Workshop on Wireless Factory Comm. Sys.*, pages 11-16, Barcelona, October 1997.
- [10] R. P. Kosowsky, I. M. Jacobs, and K. S. Gilhousen. ARNS: A new link layer protocol. In *Proceedings IEEE MILOCOM 88*, pages 515-519, September 1988.

- [11] P. R. Kumar. New technological vistas for systems and control: The example of wireless networks. *IEEE Control Systems Magazine*, 21:24-37, February 2000.
- [12] Chañe L.Fullmer and J.J.Garcia-Luna-Aceves. Solutions to hidden terminal problems in wireless networks. In *Proceedings of ACM SIGCOMM '97*, Cannes, France, September 1997.
- [13] M.Joa-Ng and I.Lu. Spread spectrum medium access protocol with collision avoidance in mobile ad-hoc wireless networks. In *Proceedings IEEE INFOCOM 99*, San Francisco, California, April 1999.
- [14] P.Gupta and P.Kumar. The capacity of wireless networks. *IEEE Transactions on Information Theory*, IT-46:388-404, March 2000.
- [15] P.Gupta, R.Gray, and P.R.Kumar. An experimental scaling law for ad hoc networks. Univ. of Illinois at Urbana-Champaign, May 2001.
- [16] V.Bharghavan, A.Demers, S.Shenker, and L.Zhang. MACAW: A medium access protocol for wireless LANs. In *Proc. of ACM SIGCOMM '94ACM*, August 1994.
- [17] Z.J.Haas and J.Deng. Dual busy tone multiple access (DBTMA): A medium access control for multihop networks. In *IEEE Wireless Communications and Networking Conference 1999*, pages 21-24, New Orleans, LA, September 1999. WCNC'99.
- [18] Z.Tang and J. J.Garcia-Luna-Aceves. Hop reservation multiple access (HRMA) for multichannel packet radio networks. In *Proceedings of the IEEE IC3N'98*. Seventh International Conference on Computer Communications and Networks, October 1998.
- [19] Z.Tang and J.J.Garcia-Luna-Aceves. A protocol for topology-dependent transmission scheduling in wireless networks. In *Proc. of IEEE WCNC*, New Orleans, LA, 1999.

THIS PAGE BLANK (USPTO)